

Image Processing - CS 474  
Programming Assignment 1  
Anthony Silva  
September 29, 2025

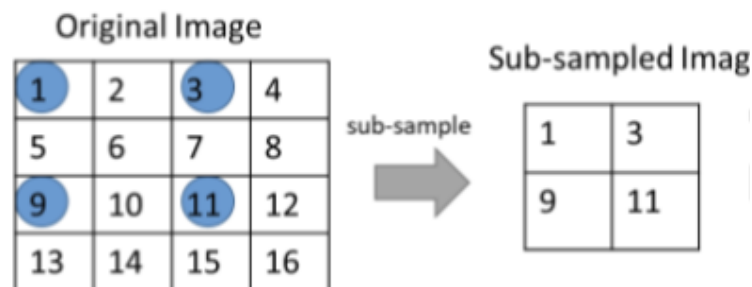
*"I declare that all material in this assignment is my own work except where there is clear acknowledgment or reference to the work of others. I understand that both my report and code may be subjected to plagiarism detection software, and fully accept all consequences if found responsible for plagiarism, as explained in the syllabus, and described in UNR's Academic Standards Policy: UAM 6,502."*

- Anthony Silva

## Theory

There are a large family of transformations applicable to images to produce certain effects for use in various domains. This report serves as an introductory exploration of point processing transformations in the spatial domain through implementations of Image Sampling, Image Quantization, and Histogram Equalization. Point processing transformations work by the definition of a transformation function that maps pixel values from one image to a new image based on a defined logic.

Image Sampling is a lossy compression transformation of the original image that reduces an image's spatial dimensions based on a defined factor. If you sample a 256x256 image by a factor of 2, it will produce a 128x128 image. A factor of 4 would produce 64x64, a factor of 8 for 32x32, and so forth. The actual sampling logic works by choosing a pixel from the original image to represent the quadrant of that image getting compressed in the spatial reduction. Quadrants are defined by evenly spaced pixel sections based on the factor chosen. Typically, the first pixel (top left) will be chosen from a quadrant to be preserved in the compression/sampling. This figure, sourced from the assignment instructions, visually elaborates on an image sampling with a factor of 2. As shown, the image is reduced by 2 in that every 2x2 quadrant of the image is reduced by taking only the first pixel in that quadrant to be preserved after the transformation. Uneven factor reductions can be handled by just taking smaller quadrants at the edges of the image when necessary (the top left pixel will be taken anyways). This transformation ultimately serves to compress data by dropping a large amount of pixel data.

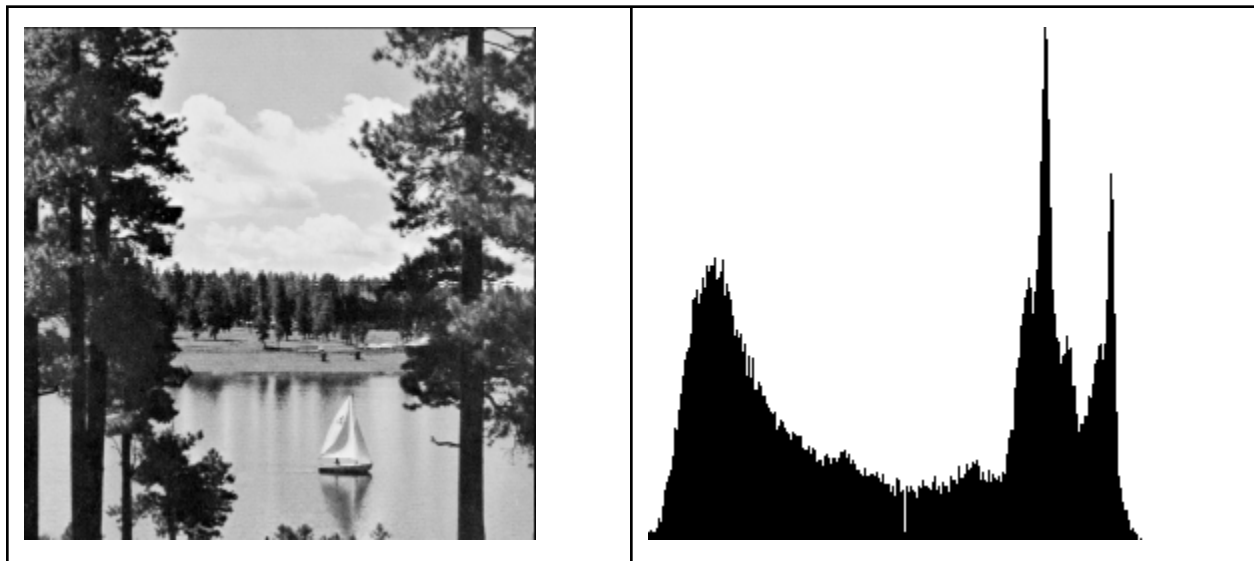


Subsampling of an image by a factor of 2.

Image Quantization is similar to Image Sampling in that it is a lossy compression transformation. However, Image Quantization differs in that it scales the pixel values of the image to lower intervals rather than the spatial resolution of the image. For grayscale images, a pixel value is typically placed within a brightness range of [0, 255] (black = 0, white = 255). A quantization by a factor of 2 would reduce this range to [0, 127], with 127 being the brightest pixel now. This transformation makes the image smaller (bitwise) at the cost of rougher gradients as there is a loss of pixel value depth.

Histogram Equalization is a transformation that attempts to increase contrast and make the image more detailed. It does this through a manipulation based on an image's gray level frequency histogram. A frequency histogram of an image shows the number of occurrences of

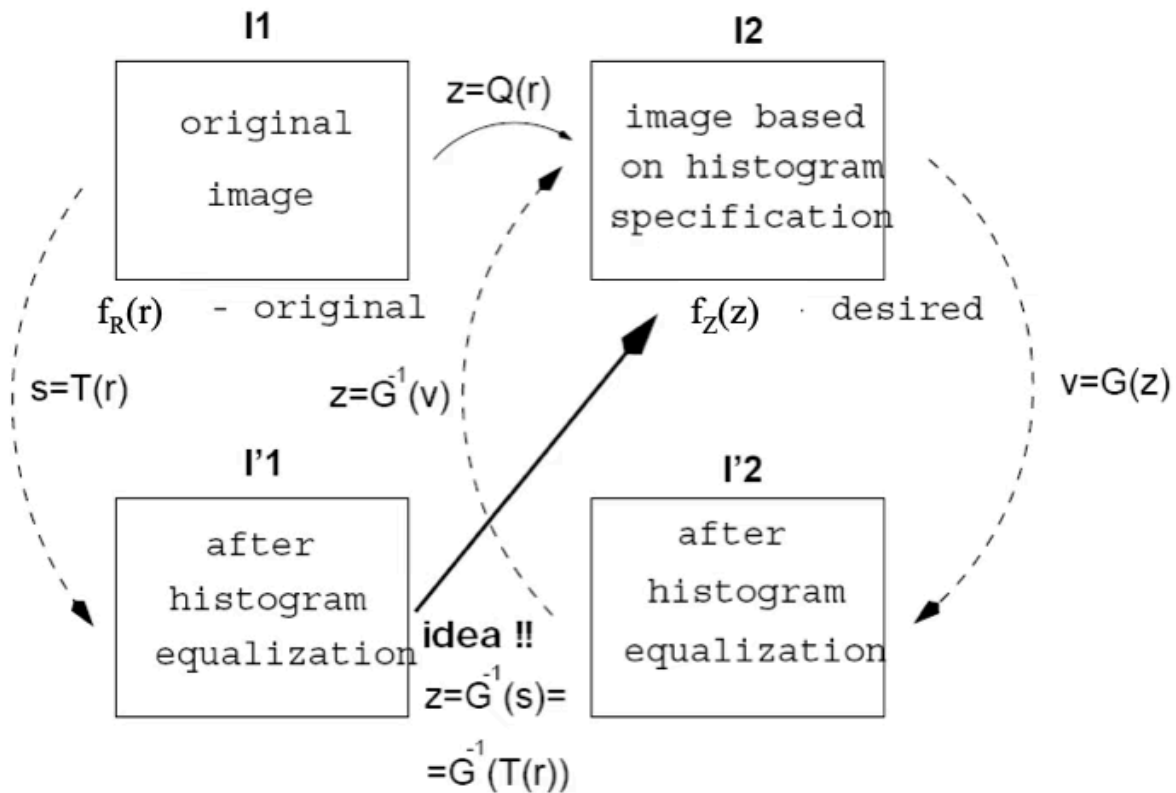
the possible pixel values in the image in histogram form. An example of an image and its frequency histogram is shown below. The benefit of histogram equalization is based on the assumption that frequency histograms with uniform distributions have clearer images than those that do not. This makes sense as a more uniform frequency distribution means that values are more evenly spaced out to allow for a better differentiation of details (higher contrast). Histogram equalization accomplishes this by creating a transform based on the specific image's histogram. The mathematical basis of this transform comes from the relationship between probability mass functions (pmf) and cumulative distribution functions (cdf). The frequency diagram of an image can be thought of as a pmf of that image's pixel values, showing the probability that a certain pixel value will occur within that image. CDFs become important in this context because in probability theory, when a random variable (like the chance that a pixel will have a certain value in an image), is transformed with the CDF of that distribution, it produces a new, uniform distribution. In the discrete case, like with these images, a perfect uniform distribution is rare, but the effects of this uniformity does provide benefits. In practice, this transformation is produced by first generating a frequency histogram of an image. Then, the cdf of that distribution can be produced (probability that a frequency is equal to or less than a pixel value). Finally, each pixel value is multiplied with its value in the cdf distribution and normalized back to the brightness range of the image to produce a more equalized frequency distribution. The resulting image of this transformation is now "equalized" and has an improved contrast.



An image and its frequency histogram.

Lastly, Histogram Specification is a technique to alter an image's frequency distribution to a specified distribution. The functionality is similar to histogram equalization, but instead of targeting a uniform distribution, you are targeting whatever distribution you specify (if you choose the uniform distribution, then it is equivalent to histogram equalization). Specification builds on equalization in that the first step of histogram specification is to in fact equalize the image you want to transform. Then, with your target frequency distribution you generate the cdf for that distribution (equivalent to another equalization transformation). Then you produce a reverse mapping (inverse function) of the target frequency cdf, where you find the intensity

value in the target distribution that corresponds to each equalized value by determining which gray level  $z$  has a cumulative probability  $G(z)$  closest to the equalized intensity. This inverse mapping  $G^{-1}$  is then applied to the equalized source image, transforming it from the uniform distribution space into the desired target distribution space, thereby completing the specification process. A diagram sourced from the class presentation on Intensity Transformations found on the class website helps visualize this process:



Histogram Specification Transformation Diagram.

## Results

I implemented these transformations within a C++ project with some functionalities and classes borrowed from the class coding resources for working with images in C++:

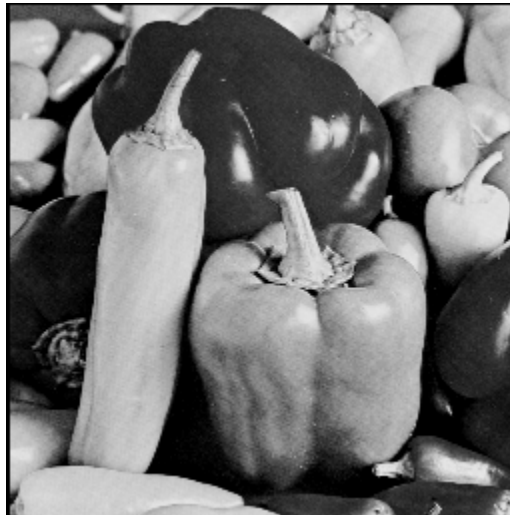
<https://www.cse.unr.edu/~bebis/CS474/>.

### Image Sampling

For Image Sampling, I used the following source images from the course website as starting points for transformations:



Lenna.pgm



Peppers.pgm

Both of these images are 256x256 pixels with a brightness range of [0, 255].

I implemented Image Sampling by looping over the source image over even intervals (based on the factor) and took the pixel at each interval to represent the sample of that quadrant. These samples reduce the spatial dimensions of the image and make it more difficult to see the differences, so I resampled the image back to 256x256. I transformed both the images on factors of 2, 4, and 8, reducing the image resolutions to 128x128, 64x64, and 32x32, respectively (which I resampled back to 256x256).



Lenna subsampled by factor of 2



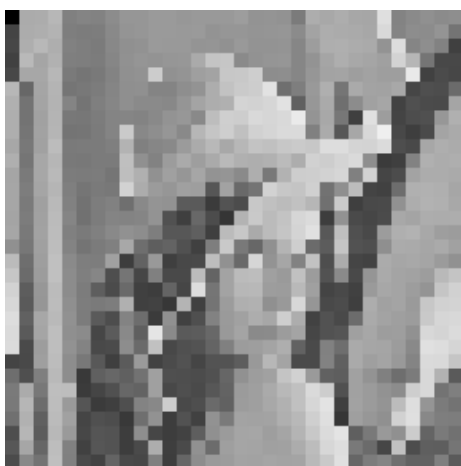
Peppers subsampled by factor of 2



Lenna subsampled by factor of 4



Peppers subsampled by factor of 4



Lenna subsampled by factor of 8



Peppers subsampled by a factor of 8

Subsampling Results.

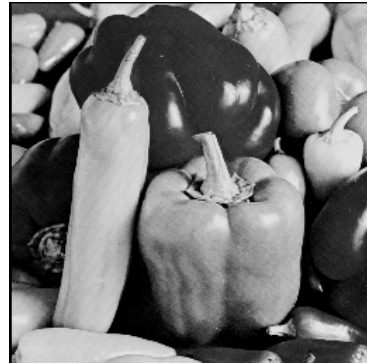
As shown, increasing the factor of subsampling reduces image quality and produces more pixelated images. This makes sense as the sampling removes pixel values and therefore information is lost and it produces a less detailed image.

### *Image Quantization*

I implemented image quantization by taking every pixel's brightness level and reducing it by the given factor through a simple division. I used the same source images as in image sampling. The results of these can be seen below:



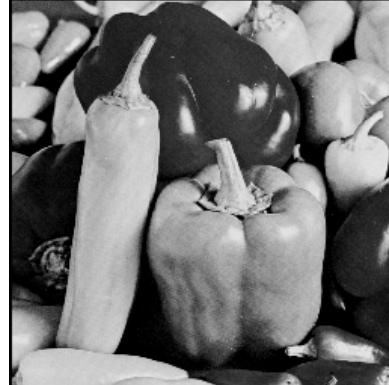
Lenna quantized by a factor of 2



Peppers quantized by a factor of 2



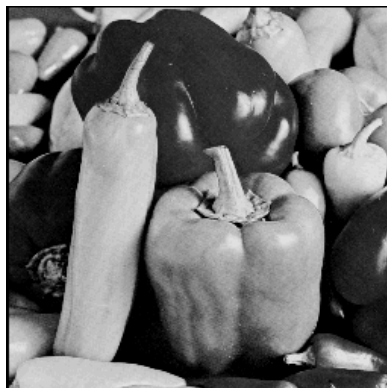
Lenna quantized by a factor of 4



Peppers quantized by a factor of 4



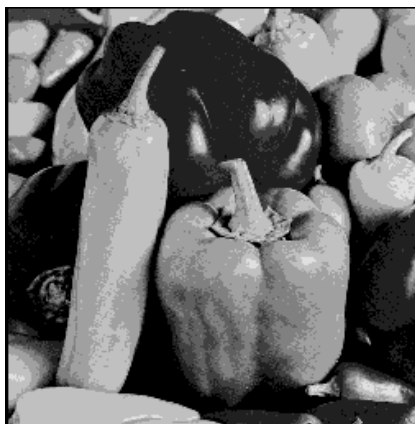
Lenna quantized by factor of 8



Peppers quantized by a factor of 8



Lenna quantized by a factor of 32



Peppers quantized by a factor of 32



Lenna quantized by a factor of 128



Peppers quantized by a factor of 128

#### Quantization Results.

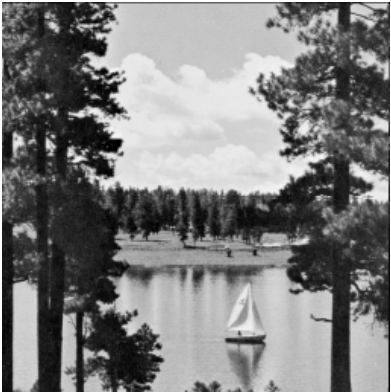


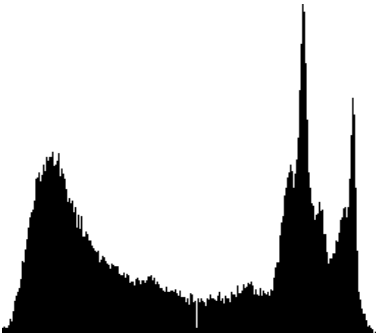

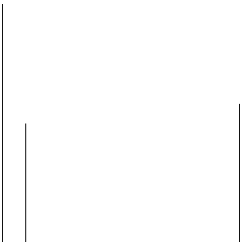
These results are interesting because data loss becomes apparent at a much later (larger factor) in comparison to image sampling. The general detail of the image is also preserved after the image quantization, with only gradients being lost with higher quantization factors (with a factor of 128, quantization effectively isolates image edges for the original image). An interesting




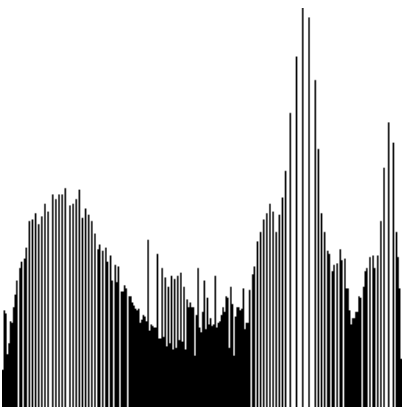
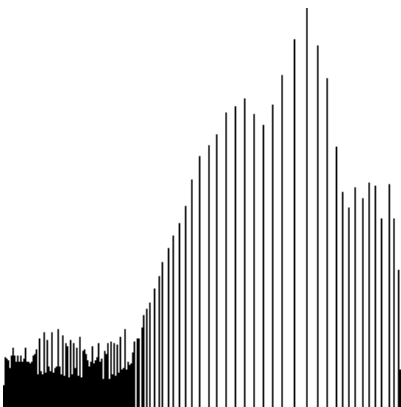



result of the algorithm is the value clipping that starts to occur (as seen in factor 32) where data loss starts to become more harshly apparent.

### *Histogram Equalization*

I implemented histogram equalization through a custom histogram class that utilized the theory behind frequency histograms and the cdf transformation to even out the distribution of the image and increase contrast. I debugged the algorithm with a test image shown in the very right column of the results below, showing an extremely basic case of the algorithm being applied. I then applied the algorithm to two more pgm images, boat.pgm and f\_16.pgm. Below is a table with the original images, their histogram distributions, the equalized images, and the frequency histograms of those equalized images.

Original Image			
Original Histogram			


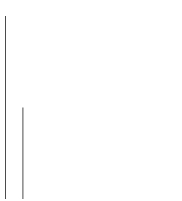


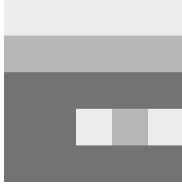
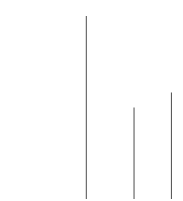

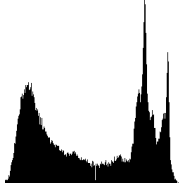

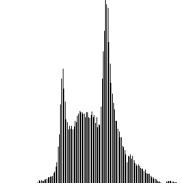

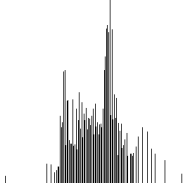

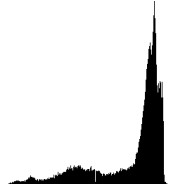
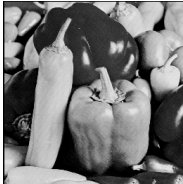
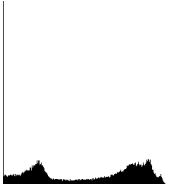

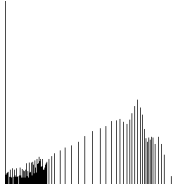
Equalized Image			
Equalized Histogram			

Equalization Results.

The differences between the original and equalized images provides great insight into how the algorithm affects the contrast of the image. It effectively takes frequency values that are heavily clustered and spreads them out. In a continuous case, this spreading out of values would result in an equalized histogram, but since our source images are so heavily discrete (256x256 with 256 pixel depth), only baby steps of that effect are seen. Regardless, the equalized images have a better contrast and more details are readily available for the viewer. In the f16 image, the clouds and mountains (which were originally tightly clustered bright values) have more depth and therefore more detail. The same can be said for the dark values of the trees and bright values of the clouds in the boat image.

## Histogram Specification

I implemented the histogram specification algorithm with a few additional functions from the histogram equalization logic that would try to find the closest frequency from the target histogram that aligns with each equalized source image gray value (basically for each gray value possible) as outlined in the theory section. To test this, I utilized the same test gray image, the boat image, and f16 image and equalized them to other images to see how they turned out.

Original Image	Original Histogram	Target Image	Target Image Histogram	Output Image	Output Image Histogram
					
					
					

Specification Results.

These results are interesting because the discreteness of the problem doesn't allow for a true histogram specification, but rather you can see artifacts of both source distributions present in the output histogram distribution. For example, the f16 image has a strong bright cluster and the target peppers distribution is much more tame with weak clustering in the dark and bright areas, but generally uniform. The resulting distribution is more uniform with an inclination towards a left skewed distribution with favor towards a brighter cluster (from the original distribution), but much more tame thanks to the target distribution's general uniformity. This shows an interesting characteristic of specifying distributions for more nuanced cases where a uniform distribution is not sufficient enough to transform an image appropriately, but instead certain specified distributions can alter more awkward source images in more useful ways.

## Discussion

This assignment provided hands-on experience with four introductory point processing transformations: Image Sampling, Image Quantization, Histogram Equalization, and Histogram Specification. Each transformation utilized a unique method of data transformation to alter images in various ways.

Image Sampling showed an intuitive way to reduce image size through spatial reduction. It was effective in reducing the visual complexity and therefore the file size of the image and it inevitably degraded the image quality in turn. Significant quality reductions occurred at a factor of 4.

Image Quantization was another dimension reduction but for pixel values rather than the image's spatial dimensions. The compression of allowable pixel ranges through quantization maintained more image fidelity at higher factors compared to image sampling. The gradual loss of gradient information through lower pixel value ranges created interesting effects with a finalization in performing essentially edge detection at the highest factor (128). This shows that image quantization has more use than just data compression and can be used as a preprocessing technique for certain computer vision tasks.

Histogram Equalization involved the most sophisticated algorithm for the cause of enhancing image contrast. The transformation effectively redistributed pixel intensities from concentrated regions to better utilize the full dynamic range. This transformation was based on the mathematical relationship of a random variable's pmf and cdf to produce a more uniform distribution. Due to the discreteness of the data, a true uniform frequency distribution could not be produced in the transformed images, but in the pursuit of such a distribution, a better image contrast was found.

Finally, Histogram Specification built on histogram equalization to transform an image's frequency distribution to not only a uniform distribution, but any distribution specified. In our limited discrete case, perfect specification alignment was not achieved, but instead artifacts of both the source image and the specified image frequency distributions were found in the resulting image frequency distributions. With these results, the usefulness of histogram specification providing a more customizable way to alter an image's contrast through frequency distribution manipulation was made apparent.

These techniques represent simple transformations in image processing with theoretical foundations that can be expanded on for more broader implementations. Understanding their principles and implementation details has helped me to have a stronger intuition for future ventures within this field, including future programming assignments within this class.